Method for organizing and compressing spatial data

U.S. Patent Application of:
Alfred Wallner

Method for organizing and compressing spatial data

Background of the Invention

1. Field of the invention

This invention relates to the field of optimization of spatial databases for functional purposes, and in particular to optimize spatial data to achieve minimal download data size for use with cartographic applications in a networked environment.

2. Discussion of prior art

A spatial database comprises topographic information in the form of shapes, lines and points encoded with geodetic coordinates, as well as sets of attributes further describing each form. Internet-based applications use the spatial database to generate bitmap images based on user input such as zip code or address on a server and transmit the map images to client devices. Bitmap-based solutions have numerous limitations, which are well known in the art. While prior art describe systems, which transfer vector data to client devices instead of bitmaps, bitmap solutions remain more efficient in terms of data transfer overhead. The initial download size of vector data is significant, and unless the user interacts repeatedly with the map, the total data amount of vector data is greater than the total data amount of bitmap images. Even if the user interacts frequently with the map and the total download size requirements for bitmap and vector data were about the same, most users prefer to have a number of shorter download wait times rather than one long download wait time. Therefore, except for a few non-mainstream applications, bitmap-based solutions are widely used in networked applications.

In navigation systems, a navigation application and spatial data are packaged and supplied as a complete system on a non-volatile storage medium. Said navigation systems may be installed in vehicles or in standalone devices. These navigation systems rely on significant computing resources such as powerful processors and large permanent storage capacities. Prior art introduces solutions, which use structuring and segmenting of spatial databases to improve data access times and navigational functionality. Said solutions are not applicable when computing resources are severely limited, as encountered on personal digital assistants and smartphones, on which one would want to have access to navigation capabilities and maps. Even when said resources are made available for car navigation systems, more powerful hardware results in higher cost for the system. More importantly still, since spatial data changes quite frequently, standalone car navigation systems will inevitably start producing out-of-date navigation instructions over time. It is therefore necessary to update the local database from time to time. Improved methods for updating said local databases have been introduced by prior art. Nevertheless, the requirement to repeatedly update data used by navigation systems remains a major inconvenience for both consumers as well as navigation system suppliers. Suppliers face substantial costs for creating and distributing the data in regular intervals, and any errors discovered after storing data sets on non-volatile media are costly to fix. To address the above problems, it is desirable to keep frequently changing spatial data on a central server and use wireless transmission networks to deliver navigation functionality to remote devices. Navigational functions such as route calculation and driving directions are performed on the server, making it also easier to integrate real-time road traffic condition data. Driving directions or maneuver instructions are text-based and relatively small in terms of data size, allowing for fairly quick wireless data transmission. On the other hand, the ability to provide graphical, cartographic map display introduces much larger data size overhead. Given the data transfer rates of wireless networks presently and during several years to come, users would experience unacceptably slow performance for map display functionality on remote devices.

The objective of this invention is to introduce a new spatial database system, which reduces the data size, makes it possible to download data in small increments as needed, and which can be used with applications such as navigation systems, for which vector-based functionality is needed.

Summary of the Invention

The primary object of the invention is to provide a compressed spatial database system, which enables incremental and efficient download of spatial vector data over a network. Another object of the invention is to introduce a location-relevant naming system so that software running on network client devices can efficiently compute data segment file names depending on user interaction with a map or device-supplied location data (e.g. GPS). A third object of the invention is to enable combined online and offline operation capability of a digital map display system. Another object of the invention is to provide server-independent map display capability based on GPS location input. A further object of the invention is to introduce a system allowing updating of dynamic location content without having to retransmit redundant map data. Yet another object of the invention is to enable map centering despite using a segmented data system.

Other objects and advantages of the present invention will become apparent from the following descriptions, taken in connection with the accompanying drawings, wherein, by way of illustration and example, an embodiment of the present invention is disclosed.

In a preferred embodiment of the present invention, a method for organizing and compressing spatial data comprises the steps of parsing a spatial database, separating topographic from attribute information, segmenting the data into rectangles, eliminating subsets of the data points, further reducing the data size by converting the data from a real number format to an integer format, generating location-relevant file names for each of the rectangles and storing the files in permanent storage space. In accordance with a preferred embodiment of the present invention, map display client software computes data file names based on user interactions or device-supplied location information (GPS), fetches the computed file names from a remote server, combines data from several data files to produce an in-memory map image and draws the image on the display screen.

Brief Description of the Drawings

The drawings constitute a part of this specification and include exemplary embodiments to the invention, which may be embodied in various forms. It is to be understood that in some instances various aspects of the invention may be shown exaggerated or enlarged to facilitate an understanding of the invention wherein:

FIG 1 is a schematic block diagram illustrating the steps to generate a compressed spatial database;

FIG 2 is a sample view of a road segment before and after applying a data size reduction algorithm;

FIG 3 is a schematic block diagram illustrating the algorithms used for data conversion as well as data segment naming;

FIG 4 is a schematic block diagram showing the different components and interactions of a network-based map display system;

FIG 5 illustrates which file names are computed by the map display program given a geodetic coordinate;

Detailed Description of the Preferred Embodiments

Detailed descriptions of the preferred embodiment are provided herein. It is to be understood, however, that the present invention may be embodied in various forms. Therefore, specific details disclosed herein are not to be interpreted as limiting, but rather as a basis for the claims and as a representative basis for teaching one skilled in the art to employ the present invention in virtually any appropriately detailed system, structure or manner.

FIG 1 shows the steps involved in organizing and compressing a spatial database 10. A spatial database comprises topographic information in the form of polygons, lines and points expressed in a geodetic coordinate system (longitude and latitude), and a set of attributes related to the topographic data entities. It is well known in the art that functions using spatial data such as routing or map display only require a subset of the entire data set. For instance, routing does not need to know about lakes and rivers, and map display does not need to know about road turn restrictions and speed limits. It is therefore possible to optimize data sets for use with a particular function. The purpose of this invention is to optimize the data set for map display functionality. The optimization process comprises a number of steps, which are described in more detail below. The initial step 11 consists of parsing the geographic database and extracting all the data for a pre-determined set of features and geographic area. The set of features comprises roads, railways, airports, rivers, lakes, shore lines, parks, points of interest and possibly others, depending how feature-rich the final map display is intended to be. In the preferred embodiment, the pre-determined geographic area is a rectangle of 1° longitude and 0.5° latitude, which will be referred to as a level 2 segment.

The parsed data from step 11 is segmented into topographic components and attribute components in step 12. Attribute information is highly redundant and is therefore an obvious compression target. Attribute information is consolidated using a simple attribute pointer or index mechanism. The topographic and attribute components are then further segmented based on location in two steps. In the preferred embodiment, the first segmentation evenly divides the area into an 8x8 grid. Each topographic data entity along with its attributes of each selected feature is assigned to one of the 64 segments. If the data entity is a polygon or a line and falls into several segments, the data entity is decomposed into two or more pieces using mathematical line and polygon splitting algorithms, and each piece is assigned to the correct segment. The resulting data segments are referred to as level 1 segments and are stored in non-volatile memory. Another segmentation is performed on the parsed data from step 11, this time dividing the area into a 64x64 grid. The resulting 4096 data segments are referred to as level 0 data segments, which are also stored in non-volatile memory.

In step 13, reduction algorithms are performed on level 1 and level 2 data. In the preferred embodiment, level 1 data covers a geographic area of 1/8° longitude by 1/16° latitude. When a map picture is generated for such an area, it is neither desirable nor practical to show all the details, especially when the map picture is shown on a small screen. For instance, it is desirable to show only the main roads, while suppressing the smaller roads. The reduction algorithm used in the preferred embodiments takes these facts into consideration. Two types of data reductions are performed. First, some topographic features such as secondary roads are completely eliminated. Second, the resolution of the remaining topographic features is reduced. Many road data entities contain a number of data points, as shown in FIG 2, which can be safely eliminated without affecting much the overall geometry of the line or polygon. For instance, the algorithm used in the preferred embodiment eliminates every other data point, provided that the angle between the two lines connecting the point to its adjacent points does not exceed 'n' degrees. Block 20 in FIG 2 shows a road segment consisting of data points 21 through 26. Block 27 in FIG 2 shows the same road after two data points, 22 and 25, have been eliminated using the above algorithm. It should be obvious from this example that eliminating these data points did not significantly change the overall shape of the road. Furthermore, it should be noted that the map resolution at this level is fairly low, meaning that data points appearing on a display screen are very close together or even overlapping. Therefore, eliminating data points as described will have no effect on what the viewer sees. After processing all 64 level 1 data files in this way, the level 2 data file is processed in similar fashion. Even more topographic features are completely eliminated. For instance, all roads except for freeways and highways are eliminated, as well as parks, points of interest and possibly other features. Resolution of the remaining topographic features is reduced even further than for level 1 data, for instance by applying the algorithm several times to the data set.

Integer conversion as referred to in block 15 of FIG 1 has two advantages. It reduces the data size by at least a factor of two, and it improves processing speed on potentially slow devices. Every geodetic coordinate is broken into two components: an offset and a value. The offset may be an aggregation of multiple offsets, but it always represents the top-left corner of a given rectangle. For any level 2 data segment (Lx-min, Lx-max, Ly-min, Ly-max), where Lx-min stands for minimum longitude, Lx-max for maximum longitude, Ly-min for minimum latitude and Ly-max for maximum latitude, the following formulas are used to compute offsets and values for each data point (x, y), where x is the longitude and y the latitude:

X2-offset = Lx-min

Y2-offset = Ly-max

X2-value = N * (x – x2-offset)

Y2-value = N * (y2-offset - y)

N = upper limit of valid integer values (50000 in the preferred embodiment)

The formulas for computing level 1 offsets and values are:

X1-offset = Abs((x – x2-offset) / ((Lx-max – Lx-min) / K))

Y1-offset = Abs((y2-offset – y) / ((Ly-max – Ly-min) / K))

X1-value = K * N * (x1 - x2-offset – x1-offset)

Y1-value = K * N * (y2-offset – y1-offset – y1)

K = segment divisor (8 in the preferred embodiment for level 1 segments)

The formulas for computing level 0 offsets and values are the same as for level 1, except that K equals 64 in the preferred embodiment.

The example shown in FIG 3 applies the above formulas to convert the geodetic coordinates 37.308805 and -122.843710 in block 30 to level 1 integers 1278 and 12516 respectively in block 35.

Level 2 offsets are shown in 31 and 32, while level 1 offsets are shown in 33 and 34. In the preferred embodiment of this invention, the upper limit N is set to 50000, but it could be a different number. The number should not exceed 65536 or 2^16, allowing it to be stored as a 2 byte integer (a short). The number should not be too low, which would result in a loss of spatial accuracy, because several real numbers would map to the same integer. The loss of accuracy is about 1 meter as implemented in the preferred embodiment of this invention.

Once a data segment has been processed and all real numbers converted to integers, a file name is assigned to the data segment as the last step in block 15 of FIG 1. Since the computed integer values are only distance values from a given base value or offset, they are not reversible to the original real number value without the offset. A simple and efficient way to supply the necessary offset values is to make them part of a file name. As shown in the example of FIG 3 block 36, a level 1 segment file name is comprised of a total of 4 numbers representing the 4 offsets used to compute integer values for that segment, as well as a letter to indicate the level, the letter 'b' representing level 1. The first number in 36 represents the level 2 latitude offset and the second number in 36 represents the level 2 longitude offset. The third number in 36 represents the level 1 latitude offset, and the fourth number in 36 represents the level 1 longitude offset.

In order to simplify computing requirements, a new geodetic coordinate system is introduced. The North Pole of the earth is at coordinate (0,0) and the South Pole is at (360,360). Unlike in the standard coordinate system, no negative values are used. Every latitude degree in the standard coordinate system corresponds to 2 latitude degrees in the new system. The conversion from the standard to the new coordinate system is accomplished as follows:

New latitude = 90 – old latitude * 2

New longitude = old longitude when range is 0° to 180°

New longitude = 180 + (180 – old longitude) when range is -180° to 0°

In the new coordinate system, moving south and east always results in greater coordinates, while moving west and north always results in smaller coordinates, until the respective end points 0 and 360 are reached. This system significantly reduces the number of exception checking operations required by map display software when compared to the standard coordinate system.
This shows that the file name contains the offset information for the spatial data stored in the file. Thus, map display software can perform a few simple calculations to compute a file name from any geodetic coordinate, which may be supplied by GPS output. It should also be evident that the task of computing file names for data segments adjacent to a given segment is very straightforward using said file-naming system.

The following section describes how a map display program can use said file system and offer desirable functionality such as combined online/offline operation. In a typical embodiment, the map display program is installed on a wireless device such as a smartphone or personal digital assistant. As shown in FIG 4, a map display system 40 consists of several functional components. The input interface layer 44 handles communication with the user or device. A text-input component lets the user type location information such as an address, a city, a zip code or a start/end point of a trip. The input interface 44 transmits said location information over the network to a geocoding engine 48 residing on a server 47. As is well known in the art, a geocoding engine computes a geodetic coordinate (longitude/latitude) from said information. Once the input interface 44 receives said geodetic coordinate from the geocoding engine 48, it notifies the map display engine 46. Some devices may have voice recognition capabilities. Instead of typing the user speaks said location information. The input interface 44 transmits the information from the voice recognition system 42 to the geocoding engine 48, waits for an answer and forwards it to the map display engine 46. Some devices may have a GPS receiver attached to or incorporated into the device. The input interface 44 processes the GPS output and relays said output to the map display engine 46 without the need to communicate with the geocoding engine 48.

The map display engine 46 uses said geodetic coordinates received from the input interface 44 to calculate four file names. The input interface 44 also tells the map display engine 46 which data level is needed, e.g. high-resolution level 0 is appropriate when the user specified an address, while level 1 may be more appropriate when the user specified a city or zip code. As has been shown in detail in a previous section, a geodetic coordinate can be decomposed and produce a unique file name. The map display engine 46 could then request said file name from a server 47 on which all files 49 are stored. However, in the preferred embodiment, the map display engine actually computes a total of four file names. If only one file is fetched, the geodetic coordinate of interest to the user could be located somewhere near the edge of said file. It would look awkward to the user and be less informative if the point of interest is not shown at or near the center of the map display screen. The ability to center the map picture has been lost by segmenting the spatial database. The solution employed by the map display engine 46 is to fetch three additional data segment files, which are most adjacent to said geodetic coordinate. The map display engine simply determines into which area, top-left, top-right, bottom-left or bottom-right, said coordinate falls. If a point falls in the top-left quadrant of a file, as does point 54 in FIG 5, the map display program first finds file 105.237.3.1.b shown in block 53, and then also fetches the file to the top, block 51, to the left, block 52, and to the top-left, block 50. After fetching all 4 files from the server 47, the map display engine combines the data of the 4 files using simple offset calculations before drawing the map picture to the screen. Said geodetic coordinates can now be displayed fairly close (within 25%) of the screen center. An even better center approximation could be achieved by using nine files. Perfect centering can be achieved by not showing a map picture of the entire available data, but instead generate a slightly zoomed-in map picture centered at said coordinate.

One objective of the invention is to provide a flexible mapping system in the sense that the map display system can function online as well as offline. Offline functionality is desirable because it offers the highest speed, since the data is accessed from local storage. The map display engine

46 gives users several options to enable offline capability. Users can select a city or zip code and download all data files for said city or zip code. Furthermore, users can reserve a certain amount of local disk space to be allocated for map data caching. When caching is enabled, the map display engine 46 automatically stores downloaded files on the local disk. As the cache fills up, new data files replace the least frequently accessed data files. A different caching algorithm, for instance based on last accessed time stamps, could be used as well. When the user has selected caching or preloading of data, the map display engine 46 always first scans the local disk space and, if available, loads data files from local space into memory instead of downloading said files from a remote server. Local caching is very useful when users frequently request the same maps. For instance, a user may want to check road traffic conditions on a daily basis. In this case, only updated traffic information such as traffic incident locations or traffic speed maps (a list of measured traffic speeds at different locations) needs to be downloaded. Said updated traffic information can be displayed on a map, which is generated from the map display engine 46 using local map data. Said offline/online capability offers optimal performance for frequently used maps as well as great flexibility regarding local storage capacities of different devices.

While the invention has been described in connection with a preferred embodiment, it is not intended to limit the scope of the invention to the particular form set forth, but on the contrary, it is intended to cover such alternatives, modifications, and equivalents as may be included within the spirit and scope of the invention as defined by the appended claims.